



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/722,294	11/25/2003	Barrie Richard Timpe	SYS 049 PA	9093
29673	7590	10/16/2007	EXAMINER	
STEVENS & SHOWALTER LLP 7019 CORPORATE WAY DAYTON, OH 45459-4238			NAM, HYUN	
ART UNIT		PAPER NUMBER		
2184				
MAIL DATE		DELIVERY MODE		
10/16/2007		PAPER		

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)
	10/722,294	TIMPE ET AL.
	Examiner Hyun Nam	Art Unit 2184

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --
Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1) Responsive to communication(s) filed on 14 August 2007.
 2a) This action is **FINAL**. 2b) This action is non-final.
 3) Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4) Claim(s) 1-29 is/are pending in the application.
 4a) Of the above claim(s) _____ is/are withdrawn from consideration.
 5) Claim(s) _____ is/are allowed.
 6) Claim(s) 1-29 is/are rejected.
 7) Claim(s) _____ is/are objected to.
 8) Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9) The specification is objected to by the Examiner.
 10) The drawing(s) filed on _____ is/are: a) accepted or b) objected to by the Examiner.
 Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).
 Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
 11) The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12) Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
 a) All b) Some * c) None of:
 1. Certified copies of the priority documents have been received.
 2. Certified copies of the priority documents have been received in Application No. _____.
 3. Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) Notice of References Cited (PTO-892)
 2) Notice of Draftsperson's Patent Drawing Review (PTO-948)
 3) Information Disclosure Statement(s) (PTO/SB/08)
 Paper No(s)/Mail Date _____

4) Interview Summary (PTO-413)
 Paper No(s)/Mail Date. _____

5) Notice of Informal Patent Application

6) Other: _____

Claim Rejections - 35 USC § 102

The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

(e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

Claims 1-25 and 27-29 are rejected under 35 U.S.C. 102(b) as being anticipated by Fung (U.S. Patent 4,535,420), hereinafter Fung '420.

Referring to claim 1, Fung '420 teaches, as claimed, a queuing system comprising:

a queue (see Fig. 1, Controller Input Buffer RAM) having a plurality of addressable storage locations (RAM) associated therewith;

queue logic (see Fig. 1; Note, entire figure represents queue logic) to control write operations to said queue (see Fig. 4, Computer Bus 15), said queue logic operatively configured to write data events (see Fig. 1, MUX 17, Device Address PROM 18, Device Address Configuration Switch, IUAX signal, DRY/FRY signal,

Bus 15, and Computer 16) to said queue in a re-circulating sequential manner (see Column 5, Line 67; and Column 6, Lines 1-2) irrespective of whether previously stored data has been read out (see Column 6, Lines 2-5);

a current event counter (see Fig. 1, RAM Access Counter 13) updated by said queue logic to keep track of a count value that corresponds to the total number of data events written (see Column 8, Lines 63-65) to said queue, said current event counter capable of counting an amount (see Column 2, Lines 22-24; Note, a 16x16 bit memory configuration requires 16 bit counter) that is greater than said plurality of addressable storage locations (see Column 4, Line 38; Note, a 16 bit counter is capable of exceeding number of addressable storage in four 4x16 memory configurations);

read logic (see Fig. 1, RAM Access Counter 13, RAM Address Holding Register 12, Tri-State Drivers 14, MUX 11, Bus 21, Bus 15, and Computer 16) operatively configured to read data events from said queue according to a prescribed manner (see Abstract, Lines 8-11), said read logic further communicably coupled to said current event counter for reading said count value stored therein (see Column 3, Lines 41-42); and

a read pointer (see Fig. 1, MUX 11 Output) controlled by said read logic that relates to where in said queue data is to be read from said queue, wherein said

read logic can read from said queue based upon said read pointer independently of write operations (see Fig. 1, MUX 17, Device Address PROM 18, Device Address Configuration Switch, IUAX signal, DRY/FRY signal, Bus 15, and Computer 16) to said queue (see Column 3, Lines 63-65).

As to claim 2, Fung '420 teaches the queuing system according to claim 1, further comprising a previous event counter (see Fig. 1, RAM Access Counter 13) that is controlled by said read logic that relates to a previous value of said current event counter (see Fig. 1, RAM Access Counter 13; Note, the Counter 13 counts previous and current events) at the time of a previous read operation on said queue by said read logic.

As to claim 3, Fung '420 teaches the queuing system according to claim 2, wherein said read logic is further operatively configured to determine whether data has been lost in the queue due to queue overflow (see Column 3, Lines 50-58) based upon a comparison of a current value (see Column 3, Lines 45-46) of said current event counter and said previous value (see Column 3, Lines 40-43) stored in said previous event counter.

As to claim 4, Fung '420 teaches the queuing system according to claim 2, wherein said read logic is further operatively configured to perform read operations according to a first prescribed manner (see Column 3, Lines 52-53) when no queue overflow is

detected (see Column 3, Lines 47-51), and said read logic performs read operations from said queue according to a second prescribed manner (see Column 3, Lines 56-58) when queue overflow is detected (see Column 3, Lines 53-55).

As to claim 5, Fung '420 teaches the queuing system according to claim 2, wherein said read pointer is derived directly from a predetermined number of the lowest order bits of said previous event counter (see Fig. 2, Counter 13 Q_A-Q_D output).

As to claim 6, Fung '420 teaches the queuing system according to claim 1, wherein a write pointer (see Fig. 1, MUX 11 Output) is derived directly from a predetermined number of the lowest order bits of said current event counter (see Fig. 2, Counter 13 Q_A-Q_c output; Note, lowest 3 bits derives count value).

As to claim 7, Fung '420 teaches the queuing system according to claim 1, wherein said queue logic stores for each data event, a combination of a sequence number derived from the value of said current event counter value (see Fig. 2, Counter 13, CK Input) and said data element (see Fig. 2, Counter 13 A, B, C, D Input tied to GND or logic zero) in said queue.

As to claim 8, Fung '420 teaches the queuing system according to claim 1, wherein said user communicates with said queue through an intermediate interface (see Fig. 1, Computer 16).

As to claim 9, Fung '420 teaches the queuing system according to claim 1, wherein said read logic is associated with a plurality of users, each user comprising:

read logic operatively configured to read information from said queue according to a prescribed manner, said read logic further communicably coupled to said current event counter for reading said count value stored therein; and

a read pointer updated by said read logic that relates to where in said queue data is to be read from said queue, wherein said read logic can read from said queue based upon said read pointer independently of write operations to said queue, wherein said read pointer and read logic of each user (see Column 2, Lines 40-45; Note, Microprocessor and Minicomputer) operates independently of one another (see Fig. 1, Buses 15, 20, and 21).

As to claim 10, Fung '420 teaches the queuing system according to claim 1, wherein said read logic further cascades data events read from said queue into a local queue for subsequent processing (see Fig. 4, Scratch Pad Memory 404).

As to claim 11, Fung '420 teaches a queuing system comprising:

a queue (see Fig. 1, Controller Input Buffer RAM) having a predetermined number of addressable storage locations (RAM);

an event counter (see Fig. 1, RAM Access Counter 13) operatively configured to sequentially update (see Fig. 2, Counter 13 CK input) a count value (see Fig. 2 Counter 13 Q_a, Q_b, Q_c, and Q_d output) stored therein each time a data event (Note, read or write event) is written into said queue, said count value capable of storing a maximum count (see Column 2, Lines 22-24; Note, 16x16 bit memory configuration requires 16 bit counter) that exceeds said predetermined number of addresses (see Column 4, Line 38; Note, a 16 bit counter is capable of exceeding number of addressable storage in 4x16 memory configuration);

a write pointer (see Fig. 1, MUX 17 output) that is derived from said count value stored in said event counter from which a select addressable storage location (see Fig. 1, Controller Input Buffer RAM 10) of said queue can be determined for temporarily storing each data event that is to be queued (see Fig. 1, Device Address PROM 18);

a read pointer (see Fig. 1, MUX 11 output) from which a desired addressable storage location (see Fig. 1, Controller Input Buffer RAM) of said queue can be identified for a read operation (see Fig. 1, RAM Access Counter 13, RAM Address Holding Register 12, Tri-State Drivers 14, MUX 11, Bus 21, Bus 15, and Computer 16);

queue logic (see Fig. 1) communicably coupled to said queue, said event counter, and said write pointer to control writing new data events (see Fig. 4, RAM 10) to said queue; and

read logic (see Fig. 1, RAM Access Counter 13, RAM Address Holding Register 12, Tri-State Drivers 14, MUX 11, Bus 21, Bus 15, and Computer 16) coupled to said queue, said event counter and said read pointer, said read logic operatively configured to read from said queue in a first manner (see Column 3, Lines 52-53) when no overflow of said queue is detected (see Column 3, Lines 47-51), and to read from said queue in a second manner (see Column 3, Lines 56-58) when overflow of said queue is detected (see Column 3, Lines 53-55).

As to claim 12, Fung '420 teaches the queuing system, according to claim 11, wherein said write pointer is encoded into said event counter (see Fig. 2, Counter 13; Note, all of the inputs to the counter and coupled logic devices to the inputs).

As to claim 13, Fung '420 teaches the queuing system according to claim 11, wherein said predetermined number of addresses of said queue is defined by the expression: $m = 2^n$, where m is the number of addresses (see Fig. 4, RAM 10, 16 slots), and n is a positive integer (see Fig. 2, four RAMs).

As to claim 14, Fung '420 teaches the queuing system according to claim 13, wherein said write pointer is defined by the lowest n bits of said event counter (see Fig. 2, Counter 13 Q_A-Q_C output; Note, lowest 3 bits defines count value).

As to claim 15, Fung '420 teaches the queuing system according to claim 14, wherein a portion of said count value is stored in said queue with each data event written thereto, said portion defined by at least one bit of said count value starting at the *n*+1 bit (see Fig. 2, Counter 13 Q_D output; Note, the 4th bit Q_D defines count value).

As to claim 16, Fung '420 teaches the queuing system according to claim 11, wherein each read from the queue is nondestructive, and each write to the queue overwrites the current content of the storage location addressed by the write pointer (see Column 6, Lines 48-60; Note, each write operation reused the one slot available in the RAM and there was no need for deleting the content of the slot when it will be rewritten).

As to claim 17, Fung '420 teaches the queuing system according to claim 11, wherein said first manner of reading from said queue comprises reading from said queue in a first in, first out manner (see Column 6, Lines 48-60; Note, FIFO with queue depth of one) such that a list sequential read follows a temporal aging from the oldest to the newest events in said queue and said second manner of reading from said queue comprises reading from said queue in a manner that reads the most recent events

written to the queue first (see Column 6, Lines 48-60; Note, read occurs soon as it is written).

As to claim 18, Fung '420 teaches the queuing system according to claim 11, wherein said second manner of reading from said queue comprises setting said read pointer to said write pointer prior to initiating a read operation (see Column 6, Lines 48-60; Note, it is inherent that RAM 10 is addressed before read or write operation begins).

As to claim 19, Fung '420 teaches the queuing system according to claim 11, wherein said second manner of reading from said queue comprises reading in a first direction (see Column 3, Lines 55-58) for a predetermined portion of a read cycle, and reading in a second direction (see Column 3, Lines 58-62) for a remainder portion of said read cycle.

As to claim 20, Fung '420 teaches the queuing system according to claim 11, wherein said second manner of reading from said queue comprises setting said read pointer to a position a predetermined number of addresses in a direction opposite to said write pointer, and beginning a read cycle wherein a plurality of data events are read in the direction of write operations to said queue (see Column 10, Lines 50-55).

As to claim 21, Fung '420 teaches the queuing system according to claim 11, wherein said read logic further comprises a previous event counter that keeps track of a

representation of said count value of said event counter at the time of a previous read operation (see Column 3, Lines 44-47).

As to claim 22, Fung '420 teaches the queuing system according to claim 21, wherein said read counter is derived from the lowest order bits of said previous event counter (see Fig. 2, Counter 13 Q_A-Q_C output; Note, lowest 3 bits derives count value).

As to claim 23, Fung '420 teaches a method of queuing data comprising:

defining a queue (see Fig. 1, Controller Input Buffer RAM) having addressable storage locations (RAM) associated therewith;

keeping track of a current count value (see Fig. 1, RAM Access Counter, Tri-State Drivers 14, and Bus 20; Note, current count value available at the Bus 20) that corresponds to the total number of data events written to said queue, said current count value capable of counting an amount (see Column 2, Lines 22-24; Note, 16x16 bit memory configuration requires 16 bit counter) that is greater than the number of said addressable storage locations (see Column 4, Line 38; Note, a 16 bit counter is capable of exceeding number of addressable storage in 4x16 memory configuration);

keeping track of a write pointer (see Fig. 1, MUX 17 output) that corresponds to a position (Address) in said queue for a write operation (see Fig. 2, Gate Logic 35 and RAM 10-1 WE or Write Enable input) thereto;

writing data events (see Fig. 1, MUX 17, Device Address PROM 18, Device Address Configuration Switch, IUAX signal, DRY/FRY signal, Bus 15, and Computer 16) to said queue in a re-circulating sequential manner (see Column 5, Line 67; and Column 6, Lines 1-2) irrespective of whether previously stored data has been read out (see Column 6, Lines 2-5); and

for each user (see Column 2, Lines 40-45; Note, Microprocessor and Minicomputer) associated with said queue:

keeping track of a previous count value that corresponds to said current count value (see Column 3, Lines 44-47) at the time of a previous access to said queue thereby; and

reading from said queue according to a prescribed manner (see Column 3, Lines 52-53).

As to claim 24, Fung '420 teaches the method according to claim 23, wherein at least one user reads from said queue according to a first prescribed manner (see Column 3,

Lines 52-53) when no queue overflow has been detected (see Column 3, Lines 47-51), and reads from said queue according to a second prescribed manner (see Column 3, Lines 56-58) when overflow has been detected (see Column 3, Lines 53-55).

As to claim 25, Fung '420 teaches the method according to claim 24, wherein queue overflow is detected if the difference between said, current count value and said previous count value is greater than a total number of said addressable storage locations (see Fig. 2, Counter 13 CO or Carry Output; Note, it is inherent that counter has carry output when overflow of counter occurs which indicates that current counter exceeded its addressable storage).

As to claim 27, Fung '420 teaches the method according to claim 24, wherein a select user reads from said queue comprising:

reading said current count value (see Column 3, Lines 45-46);

reading said previous count value (see Column 3, Lines 46-47);

comparing said current count value to said previous count value to determine whether overflow has occurred to the queue with respect to said user (see Column 3, Lines 46-53);

if no overflow is detected:

reading a queued data (see Column 3, Line 62) event based upon said read pointer;

updating said read pointer based upon said first predetermined manner (see Column 3 Line 52-53); and

updating said previous event counter value; and

if overflow is detected:

updating said read pointer based upon a second predetermined manner;

reading at least one queued data event based upon said read pointer;

updating said read pointer based upon said second predetermined manner; and

updating said previous event counter value.

As to claim 28, Fung '420 teaches the method according to claim 27, further comprising after each read of said queue where no overflow is detected:

determining a new current count value (see Column 3, Lines 45-46);

comparing said new current count value to said previously stored count value to determine whether overflow has occurred (see Column 3, Lines 46-53); and

if overflow is detected:

updating said read pointer based upon a second predetermined manner;

reading at least one queued data event based upon said read pointer;

updating said read pointer based upon said predetermined manner; and

updating said previous event counter value.

As to claim 29, Fung '420 teaches the method according to claim 27, wherein said read pointer is updated to a new position that corresponds to a current value of said write pointer (see Column 6, Lines 50-55; Note, after a write event a read event occurs at same address point).

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

Claim 26 is rejected under 35 U.S.C. 103(a) as obvious over Fung '420 in view of Smith et al. (U.S. Patent 4,872,110), hereinafter Smith '110.

As to claim 26, Fung '420 teaches the method according to claim 24, wherein a select user requests data events from said queue comprising:

specifying a total number of requested data events (see Column 6, Lines 46-48);

accessing said queue to obtain said data events (see Column 6, Lines 52-53);
and

queueing any extracted data events in a local queue accessible by said user (see Column 6, Lines 60-61).

Fung '420 does not expressly discloses a method wherein a select user requests data events from said queue comprising specifying a timeout period that represents the maximum amount of time said user is willing to wait for said data events.

Smith '110 does disclose a method wherein a select user requests data event comprising specifying a timeout period (see Figure 3A Timer 68 and 74) that represents the maximum amount of time said user is willing to wait for said data events (see Column 4, Line 23).

Therefore, it would have been obvious to one of ordinary skill in the art at the time of Applicant's invention to incorporate command acknowledge and timeout circuit of Smith et al. to the queue system of Fung '420.

One of ordinary skill in the art would be motivated to do because keeping track of acknowledge and timeout signal will improve software throughput and operator will be immediately alerted if problem exists.

Claims 1 and 2 are rejected under 35 U.S.C. 103(a) as obvious over Unger (U.S. Patent 3,935,563), hereinafter Unger '563 in view of Heap et al. (U.S. Patent 4,231,106), hereinafter Heap '106.

Referring to claim 1, Unger '563 teaches, as claimed, a queuing system comprising:

a queue (see Fig. 2, Footprint RAM 42) having a plurality of addressable storage locations (see Column 3, Line 68) associated therewith; queue logic (see Figure 2) to control write operations (see Fig. 2, Write/Count line 43) to said queue, said queue logic operatively configured to write data events to said queue in a recirculating sequential manner (see Column 3, Lines 5-7) irrespective of whether previously stored data has been read out;

a current event counter (see Fig. 2, Readout Counter 62) updated by said queue logic to keep track of a count value (see Fig. 2, Readout Counter 62 output) that corresponds to the total number of data events (see Fig 2, Count/Read 337 line 64) written to said queue;

read logic (see Fig. 2) operatively configured to read data events (see Fig. 2, Reference Updater 50) from said queue according to a prescribed manner (see Column 3, Line 12), said read logic further communicably coupled to said current event counter for reading said count value stored therein (see Fig. 2, Circle Counter 44 output); and

a read pointer (see Fig. 3, Selector 54 output) controlled by said read logic that relates to where in said queue data is to be read from said queue (see Fig. 2, Footprint RAM 42 and Z Register 30), wherein said read logic can read from said

queue based upon said read pointer independently of write operations (Note, the write operation does not influence the read operation) to said queue.

Unger '563 does not disclose expressly a queuing system comprising a current event counter capable of counting an amount that is greater than said plurality of addressable storage locations.

Heap '106 does disclose a queuing system comprising a current event counter capable of counting an amount that is greater than said plurality of addressable storage locations (see Abstract, Lines 3-5; Note, it is inherent that the prescribed instruction sequence executed in prescribed functions are capable of exceeding any given queue size in infinite test loop scenario).

Therefore, it would have been obvious to one of ordinary skill in the art at the time of Applicant's invention to incorporate counter technique of Heap et al. to a queuing or memory footprint system of Unger.

One of ordinary skill in the art would be motivated to do because the counting apparatus can produce a record of the occurrence of various software events or actions in the data processor (see Column 2, Lines 60-62).

As to claim 2, the modification teaches the queuing system according to claim 1, further comprising a previous event counter (see Unger '563, Fig. 2, Circle Counter 44) that is controlled by said read logic that relates to a previous value of said current event counter (see Heap '106, Fig. 1, Timing and Counting Apparatus 12) at the time of a previous read operation on said queue by said read logic.

Claim 1 is rejected under 35 U.S.C. 103(a) as obvious over Unger '563 in view of Lindsay's publication, *A Hardware Monitor Study of a CDC KRONOS System*, hereinafter Lindsay.

Referring to claim 1, Unger '563 teaches, as claimed, a queuing system comprising:

a queue (see Fig. 2, Footprint RAM 42) having a plurality of addressable storage locations (see Column 3, Line 68) associated therewith; queue logic (see Fig. 2) to control write operations (see Fig. 2, Write/Count line 43) to said queue, said queue logic operatively configured to write data events to said queue in a recirculating sequential manner (see Column 3, Lines 5-7) irrespective of whether previously stored data has been read out;

a current event counter (see Fig. 2, Readout Counter 62) updated by said queue logic to keep track of a count value (see Fig. 2, Readout Counter 62 output) that

corresponds to the total number of data events (see Fig. 2, Count/Read 337 line 64) written to said queue;

read logic (see Fig. 2) operatively configured to read data events (see Fig. 2 Reference Updater 50) from said queue according to a prescribed manner (see Column 3, Line 12), said read logic further communicably coupled to said current event counter for reading said count value stored therein (see Fig. 2, Circle Counter 44 output); and

a read pointer (see Fig. 3, Selector 54 output) controlled by said read logic that relates to where in said queue data is to be read from said queue (see Fig. 2 Footprint RAM 42 and Z Register 30), wherein said read logic can read from said queue based upon said read pointer independently of write operations (Note, the write operation does not influence the read operation) to said queue.

Unger '563 does not disclose expressly a queuing system comprising a current event counter capable of counting an amount that is greater than said plurality of addressable storage locations.

Lindsay does disclose a current event counter capable of counting an amount that is greater than said plurality of addressable storage locations (see Page 136, Section II.A.1, Line 16).

Therefore, it would have been obvious to one of ordinary skill in the art at the time of Applicant's invention to incorporate counter technique of Lindsay to a queuing or memory footprint system of Unger.

One of ordinary skill in the art would be motivated to do because the counting apparatus can produce a record of the occurrence of various software events or actions in the data processor by multiple of users. This with the memory footprint readout will accomplish comprehensive analysis of the given software (see Page 136, Section II.A.1, Lines 20-24).

Response to Arguments

Applicant's arguments filed 08/14/2007 have been fully considered but they are not deemed to be persuasive.

Regarding the 35 U.S.C. §112, second paragraph problems, Applicant's response and amendment have overcome these rejections.

Applicant argues, as to claim 1, the access counter 13 controls one of two possible sources of address information used by the RAM 10 and is utilized to implement a form of dual port memory. However, Fung fails to teach or suggest that the access counter 13 implements a current event counter updated by queue

logic to keep track of a count value that corresponds to the total number of data events written to the queue as claimed. For example, since the access counter 13 stores an address used to access the RAM, when the limits of the storage capacity of the RAM have been reached, the address stored by the access counter will merely reset to the next available storage address. That is, the access counter does not count events, but rather tracks (and increments) addresses for accessing the RAM (see Applicant's Argument, Pages 10-11).

In this regard, Fung is completely silent to the teaching or suggestion of any counter that keeps track of a count value that corresponds to the total number of data events written to the queue as claimed. For example, the passage cited by the Examiner at Col. 8, lines 63-65 explicitly states that the counter 13 merely "... provides a pointer to sequentially select the next available location ..." in the RAM4. Col. 2, lines 22-24 merely gives an exemplary size of the RAM 10 and Col. 4, line 38 illustrates how the total capacity of RAM 10 (e.g., 16x16) can be implemented using multiple physical memory devices, e.g., four 4x16 RAMS that can be combined for the total 16x16 memory. The above cited passages fail to teach or suggest at least a current event counter updated by said queue logic to keep track of a count value that corresponds to the total number of data events written to said queue as claimed. In view of the clarifying comments herein, the applicants' respectfully request that the rejection of claim 1 under 35 U.S.C. 5

102(b) and the claims that depend there from, be withdrawn (see Applicant's Argument, Page 11).

As claim 11 recites an event counter operatively configured to sequentially update a count value stored therein each time a data event is written into said queue, the arguments in support of claim 1 set out above apply by analogy to claim 11. In view of the clarifying comments herein, the applicants' respectfully request that the rejection of claim 11 under 35 U.S.C. 5 102(b) and the claims that depend there from, be withdrawn (see Applicant's Argument, Page 12, Paragraph 1).

As claim 23 recites keeping track of a current count value that corresponds to the total number of data events written to the queue, the arguments in support of claim 1 set out above apply by analogy to claim 23. In view of the clarifying comments herein, the applicants' respectfully request that the rejection of claim 23 under 35 U.S.C. 5 102(b) and the claims that depend there from, be withdrawn (see Applicant's Argument, Page 12, Paragraph 3).

The applicants respectfully submit that claim 26 is patentable over Fung in view of Smith by virtue of being dependent upon claim 23, which the applicants believe to be patentable as described in greater detail herein (see Applicant's Argument, Page 12, Paragraph 4).

Examiner disagrees with applicant. The terms in the claims 1, 11, 23, and 26, 'current event counter', 'queue logic', 'count value' and 'data events' at best has been loosely defined. The access counter 13 counts and tracks the values (i.e. address values), which are of current event that relates to data events (Note, even 'address' is called 'data' when it is stored in a buffer or a queue; however, RAM 10 shown in Figure 4 of Fung '420 illustrates that the queue holds more then 'address data'). Furthermore, the term, 'queue logic' has no structural bounds according to the claim language. Hence, entire circuit shown in Figure 1 of Fung '420 represents 'queue logic'; it counts, updates, and keeps track of count/address values using circular-queue structure. Also, the count/address value does correspond with (i.e. in contact with, associated with, or agree with) the total number of data events within the queue.

Applicant argues, as to claims 1 and 2, the Readout counter 62 in Unger is used to control an adder that determines the address of the instruction and/or operand reference that is read out from the queues. In particular, the Readout counter 62 is cleared each time data is stored in address 337, when lead 66 is active or when a master clear is active on lead 72. For a first read, the Readout counter 62 has a value of zero (0) and the oldest data is selected via the address determined by the counter 44 and the Readout counter 62. At the end of the read, the Readout counter 62 is incremented. However, after 16 reads, the sequence repeats. Thus, the Readout counter 62 is cleared again. That is, the Readout counter 62 does not keep track of a count value that corresponds to the total number of data events

written to the queue as claimed (see Applicant's Argument, Page 13, Paragraph 4).

Moreover, Unger is completely silent with regard to, and thus fails to teach or suggest any structure that is updated by queue logic to keep track of a count value that corresponds to the total number of data events written to the queue (see Applicant's Argument, Page 14, Paragraph 1).

Further, the Examiner fails to point to any teaching or suggestion in Heap that teaches or suggests a current event counter updated by queue logic to keep track of a count value that corresponds to the total number of data events written to the queue (see Applicant's Argument, Page 14, Paragraph 2).

Moreover, the Examiner fails to point to any teaching or suggestion in Lindsay that teaches or suggests a current event counter updated by queue logic to keep track of a count value that corresponds to the total number of data events written to the queue (see Applicant's Argument, Page 14, Paragraph 6).

Examiner disagrees with applicant. The term in the claims 1 and 2, 'total number of data events' at best has been loosely defined. The 'total number of data events' is represented by total number of entry in a queue at any given time. For example, when the total number of entry in the queue is 15 then the Readout counter 62 has kept track

of 15 entries. Also, all counters has finite number it can count up to without recycling; therefore, counter in the Unger '563 reads into counter as claimed.

Conclusion

THIS ACTION IS MADE FINAL. Applicant is reminded of the extension of time policy as set forth in 37 CFR 1.136(a).

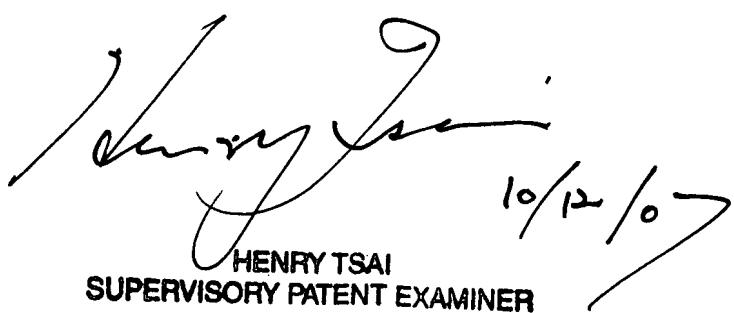
A shortened statutory period for reply to this final action is set to expire THREE MONTHS from the mailing date of this action. In the event a first reply is filed within TWO MONTHS of the mailing date of this final action and the advisory action is not mailed until after the end of the THREE-MONTH shortened statutory period, then the shortened statutory period will expire on the date the advisory action is mailed, and any extension fee pursuant to 37 CFR 1.136(a) will be calculated from the mailing date of the advisory action. In no event, however, will the statutory period for reply expire later than SIX MONTHS from the date of this final action.

Contact Information

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Hyun Nam whose telephone number is (571) 270-1725. The examiner can normally be reached on Monday through Friday 8:30 AM to 5:00 PM EST. If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Dr. Henry Tsai can be reached on (571) 272-4176. The fax phone number

for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.



10/12/07

HENRY TSAI
SUPERVISORY PATENT EXAMINER